

Logic Programming Engineering – Assignment 0

Dr.-Ing. Sascha Klüppelholz

Exercise 0.1 [Just play and see how it actually works]

Type in the following Prolog program.

```
letter(a).  
letter(b).  
letter(c).  
perm3(L1,L2,L3) :- letter(L1), letter(L2), L1\=L2, letter(L3), L3\=L1, L3\=L2.
```

- Type in the following query: “perm3(X,Y,Z).”. Interpret the output. Press “;” to generate all possible solutions. What is the intended semantics of the predicate perm3?
- Type in “trace.” to enable the trace mode and redo Task a). Try to understand what the Prolog interpreter does for the given program. You can leave trace mode with “notrace.”.
- Play around with the given program by, e.g.
 - adding new letters, removing letters, changing the order in which letters are defined,
 - swapping the order in which letters and perm3 are defined,
 - changing the order of the terms letter(·) and $L_i \neq L_j$ within perm3,
 - ...

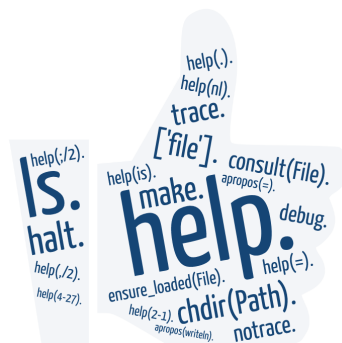
How do these changes effect the result and/or the order in which solutions are found?

- Add the following line to your program:

```
perm3((X1,X2,X3)) :- perm3(X1,X2,X3).
```

and reconsult the program. Query it with “perm3(X).” and “perm3(L,a,b).”.

- Add a predicate pal6 which is similar to perm3, but evaluates to true, if six letters form a palindrome.
- Execute and understand the following commands. Read the respective paragraph of the manual. For next week, please read Sections 2.1 (Getting started quickly) and 4.27 (Arithmetic).



Solution:

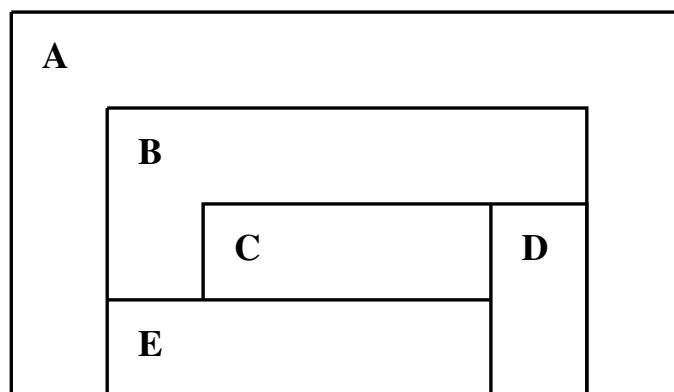
For part e) of the exercise: `pal6(X1,X2,X3,X3,X2,X1) :- letter(X1), letter(X2), letter(X3).`

Exercise 0.2 [Four colors]

Let $G = (V, E)$ be a graph, where V is a finite set of nodes and $E \subseteq V \times V$ the edge relation. A coloring of G is a total function $\text{col} : V \rightarrow \text{Colors}$ that assigns a color to each of the nodes such that neighboring nodes carry different colors, i.e., $\text{col}(v_1) = \text{col}(v_2)$ implies $(v_1, v_2) \notin E$.

The goal of this exercise is to write a Prolog program that allows for computing all colorings of a given graph G with four colors.

- Define the four colors red, blue, yellow and green as facts `color/1` in your program.
- Define the predicate `coloring(A,B)` that evaluates to true if `color(A)` is different from `color(B)`.
- Encode the following map into the predicate `graphcolor/5`.



Your program should be the conjunction of sub-goals `coloring(X,Y)` for all nodes $X, Y \in E$ and should start like this: `graphcolor(A,B,C,D,E) :- coloring(A,B), coloring(A,E), ...`

- Is there a solution for which A and C have different colors? Find this out using the following query: `“graphcolor(A,B,C,D,E), A \= C.”`
- Compute all possible four-colorings of the map above using the query `“graphcolor(A,B,C,D,E).”`. How many solutions are there?

Solution:

- ```
color(red).
color(blue).
color(yellow).
color(green).
```
- `coloring(A,B) :- color(A), color(B), A \= B.`
- ```
graphcolor(A,B,C,D,E) :- coloring(A,B), coloring(A,D), coloring(A,E), coloring(B,C),
    coloring(B,E), coloring(B,D), coloring(C,E), coloring(C,D), coloring(D,E).
```
- `graphcolor(A,B,C,D,E), A \= C.`
`false.`
- `findall((A,B,C,D,E), graphcolor(A,B,C,D,E), L), length(L, Len), writeln(Len).`
`24.`